Measurement Science for Complex Information Systems

Project Web Page is http://www.nist.gov/itl/antd/emergent_behavior.cfm

K. Mills & C. Dabrowski (computer science), J. Filliben (statistics),
F. Hunt & D. Genin (math), S. Ressler (infoViz)

image generated with http://www.wordle.net/ applied to contents of a paper entitled "Sensitivity Analysis of Koala: an Infrastructure Cloud Simulator" written by Mills, Filliben and Dabrowski

"We can capture lots of data,
but we can't always make sense of it."

David Alan Grier, computer science professor at George Washington University,
"Investing in Ignorance", *Computer Magazine,* Dec. 2010, page 15.

Measurement science is about determining what data to capture and under what conditions so that we ***can*** make sense of it.

What are complex systems?

Large collections of interconnected components whose interactions lead to macroscopic behaviors

– Biological systems (e.g., slime molds, ant colonies, embryos)
– Physical systems (e.g., earthquakes, avalanches, forest fires)
– Social systems (e.g., transportation networks, cities, economies)
– Information systems (e.g., Internet, Web services, compute grids)

What is the problem?

No one understands how to measure, predict or control macroscopic behavior in complex information systems

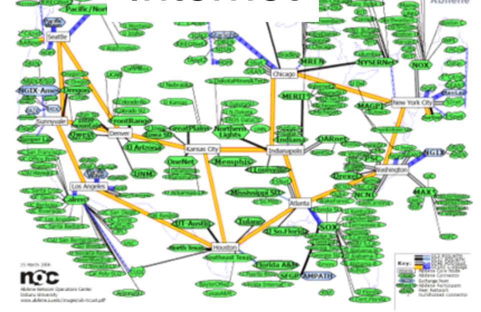
"[Despite] society's profound dependence on networks, fundamental knowledge about them is primitive. … [G]lobal communication … networks have quite advanced technological implementations but their behavior under stress still cannot be predicted reliably.… There is no science today that offers the fundamental knowledge necessary to design large complex networks [so] that their behaviors can be predicted prior to building them."

— Network Science, NRC report released in 2006

What is the new idea?

Leverage models and mathematics from the physical sciences to define a systematic method to measure, understand and control macroscopic behavior in large distributed information systems, such as the Internet and computational clouds and grids

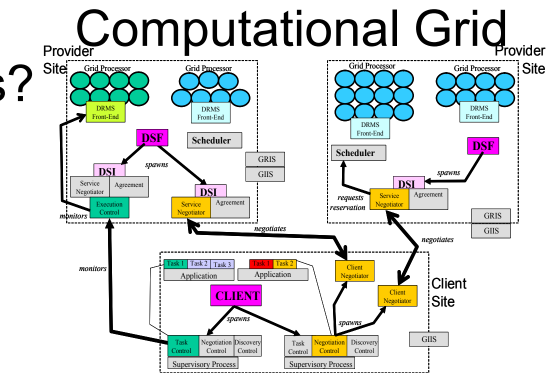
Internet



Technical Approach

- Evaluate models and analysis methods
  - Are they computationally tractable?
  - Can they reveal macroscopic behavior?
  - Can they establish causality?
- Evaluate distributed control techniques
  - Can economic mechanisms elicit desired behaviors?
  - Can biologically inspired mechanisms organize elements?
  - Can heuristics allocate resources efficiently?

Computational Grid



Hard Issues & Approaches Investigated

Hard Issues	Solutions Investigated and Evaluated
1. Model Scale	• Model restriction and parameter clustering (leading to MesoNet and Koala) • 2-level experiment designs • Orthogonal fractional factorial (OFF) experiment designs • Markov chains
2. Model Validation	• Sensitivity analysis • Key comparisons with empirical results in small topologies • Generating Markov chain models from discrete-event simulations
3. Tractable Analysis	• Correlation analysis with clustering • Principal components analysis • 10-step graphical analysis • Cluster analysis • Custom multidimensional visualizations • Exploratory interactive multidimensional visualization • Eigenanalysis of matrices
4. Causal Analysis	• Principal components analysis • Detailed measurements of model behavior • Time series analysis • Hypothesis testing • Exploratory analyses • Cut set analysis of graphs and perturbation of Markov chain models
5. Controlling Behavior	• Economic algorithms for resource allocation in computational grids • Proposed Internet congestion control algorithms • Heuristics for resource allocation in infrastructure clouds

Sensitivity Analysis of *Koala*: an Infrastructure Cloud Simulator

K. Mills, J. Filliben, C. Dabrowski
and S. Ressler

image generated with http://www.wordle.net/ applied to source code for Koala Cloud Simulator created by Mills and Dabrowski

Synopsis

Problem: Resource allocation in on-demand Clouds can be formulated as an on-line bin packing problem, where algorithms cannot always achieve optimality, implying algorithms will be heuristics.

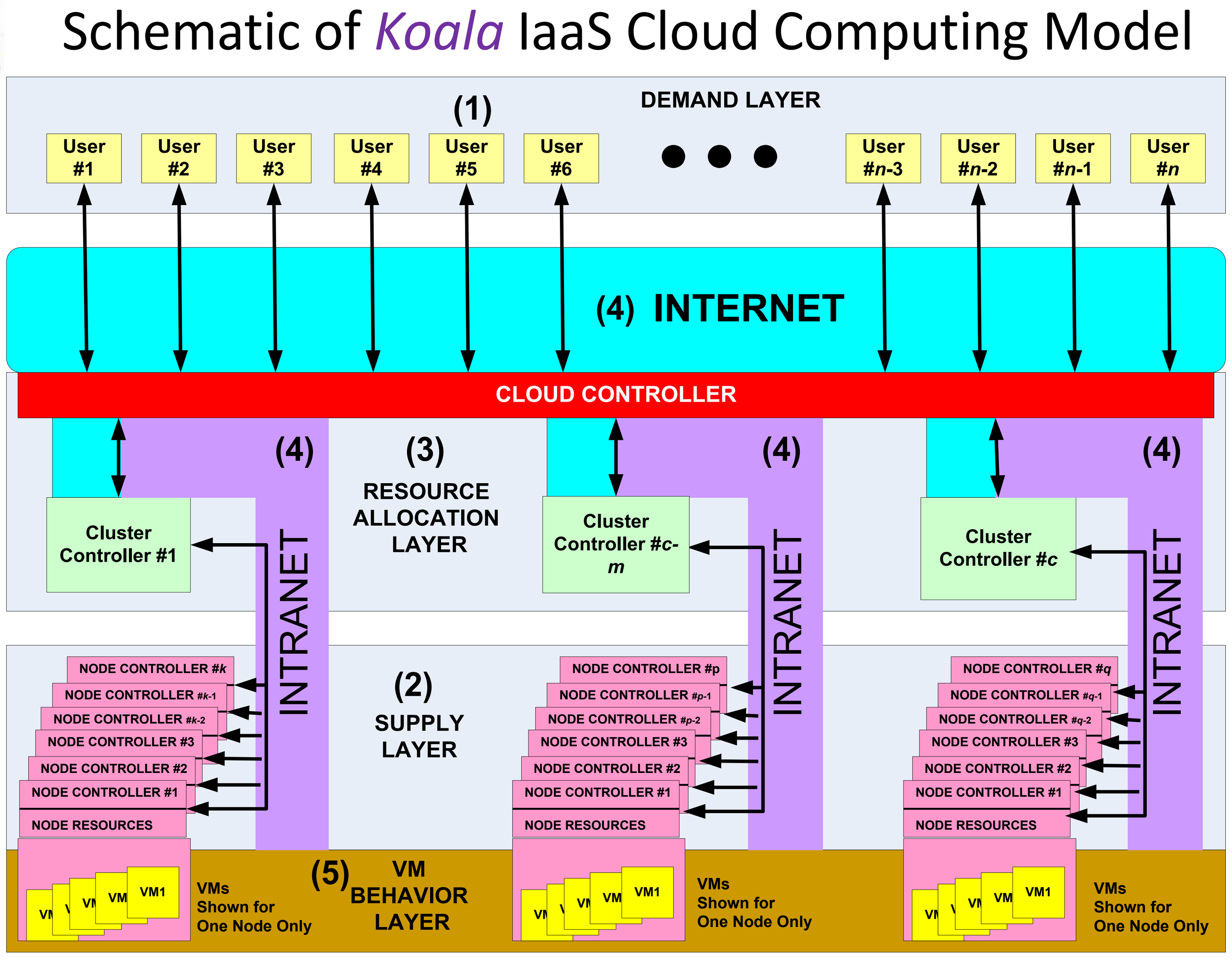
Objective: We are applying our methods to compare 18 resource allocation heuristics for on-demand infrastructure Clouds.

First steps (describing today):

(1) Formulate *Koala*, a reduced scale model created by identifying, restricting and grouping parameters
(2) Identify essential *Koala* behaviors by applying correlation analysis and clustering
(3) Identify *Koala* parameters that significantly influence essential behaviors by applying 2-Level orthogonal fractional factorial (OFF) experiment designs

Next steps (ongoing): (1) Apply 2-Level OFF design again to create comparison conditions, (2) Simulate each heuristic under created conditions, and (3) Apply multidimensional analysis techniques to identify significant patterns and causality

Schematic of *Koala* IaaS Cloud Computing Model



2-Level OFF Experiment Designs Reduce # of Parameter Combinations, While Improving Global Coverage and Minimizing Error in Effect Estimates in comparison with comparable Factor-at-a-Time (FAT) Designs

We selected two pairs of level settings (SA1 & SA2) and two system sizes (small & large)

Adopted 2-Level (2^{11-5}) "Resolution IV" OFF experiment design, requiring 64 simulations per experiment

Instantiated 4 designs, and simulated 6 repetitions (different random number seeds) with the 2 smaller designs

Required (6 x 2 + 2) x 64 = 896 simulations

	SA1-small and SA1-large		SA2-small and SA2-large	
Parameter	Plus Level	Minus Level	Plus Level	Minus Level
x1	1200 hours	600 hours	1600 hours	200 hours
x2	500 (SA1-small) 5000 (SA1-large)	250 (SA1-small) 2500 (SA1-large)	750 (SA2-small) 7500 (SA2-large)	125 (SA2-small) 1250 (SA2-large)
x3	PU1 = 0.2 PU2 = 0.2 PU3 = 0.1 PU4 = 0.1 WS1 = 0.15 WS2 = 0.07 WS3 = 0.03 PS1 = 0.1 PS2 = 0.01 MS1 = 0.1 MS3 =0.01 DS1 = 0.10 DS2 = 0.01	PU1 = 1/6 PU2 = 1/6, WS1 = 1/6 MS1 = 1/6 PS1 = 1/6 DS1 = 1/6	PU1 = 0.4 PU2 = 0.4 PU3 = 0.1 PU4 = 0.05 PU5 = 0.025 PU6 = 0.025	WS1 = 0.25 WS2 = 0.15 WS3 = 0.1 PS1 = 0.35 PS2 = 0.04 PS3 = 0.01 DS1 = 0.08 DS2 = 0.015 DS3 = 0.005
x4	8 hours (α = 1.2)	4 hours (α = 1.2)	12 hours (α = 1.2)	2 hours (α = 1.2)
x5	20 (SA1-small) 40 (SA1-large)	10 (SA1-small) 20 (SA1-large)	30 (SA2-small) 40 (SA2-large)	5 (SA2-small) 10 (SA2-large)
x6	200 (SA1-small) 1000 (SA1-large)	100 (SA1-small) 500 (SA1-large)	400 (SA2-small) 1500 (SA2-large)	50 (SA2-small) 250 (SA2-large)
x7	C22 = 1.0	C8 = 0.25 C14 = 0.25 C18 = 0.25 C22 = 0.25	C14 = 0.2 C16 = 0.2 C18 = 0.2 C20 = 0.2 C22 = 0.2	C2 = 0.1 C4 = 0.1 C6 = 0.1 C8 = 0.1 C10 = 0.1 C12 = 0.1 C16 = 0.1 C22 = 0.3
x8	Percent Allocated	Least-Full First	Percent Allocated	Least-Full First
x9	Next-Fit	First-Fit	Next-Fit	First-Fit
x10	4	1	8	1
x11	10^{-3} to 10^{-8}	10^{-4} to 10^{-9}	10^{-2} to 10^{-7}	10^{-5} to 10^{-10}

Correlation Analysis & Clustering (CAC) Reduces Dimensionality

We identified an **8-dimensional response space** within the 40 responses

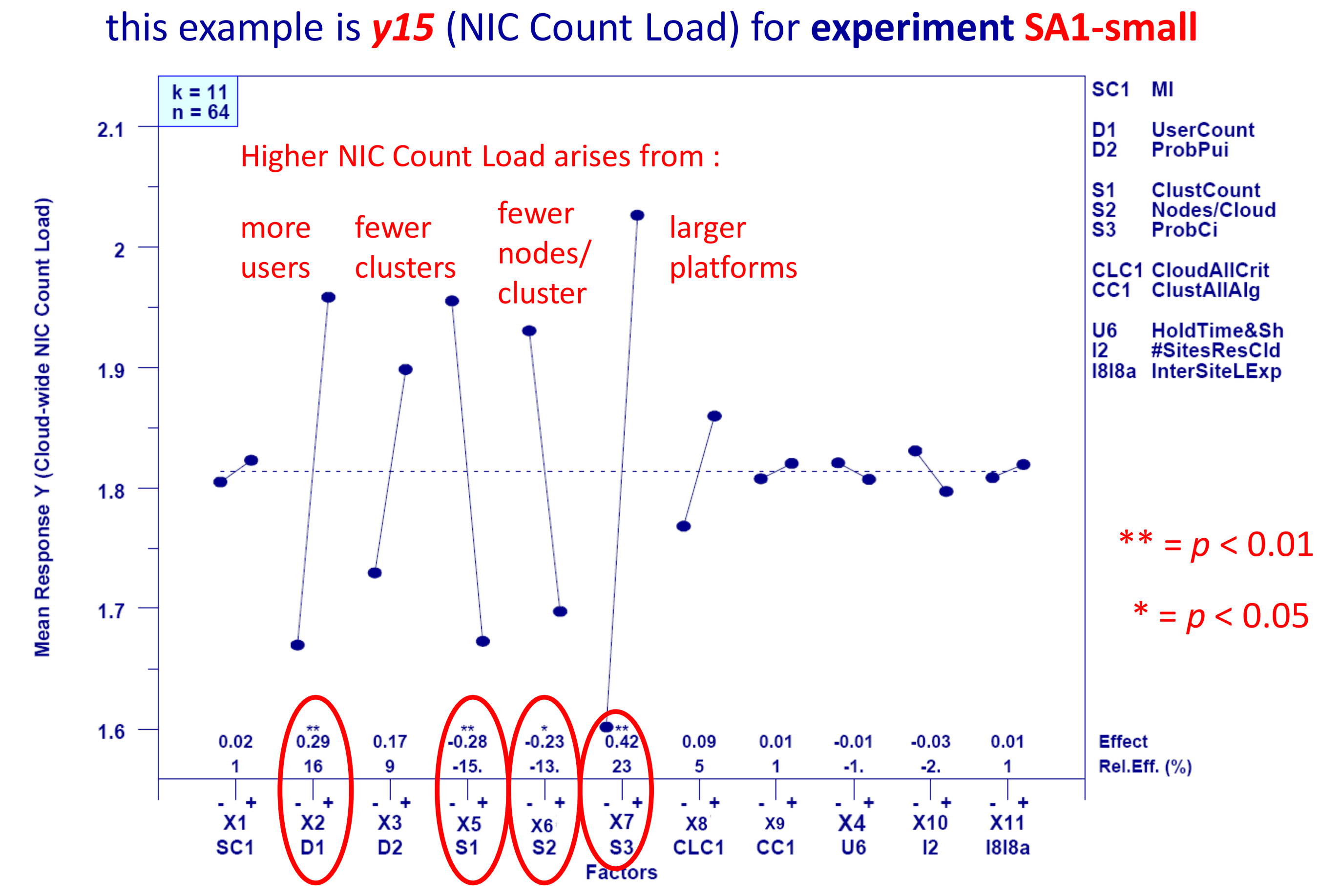
1. Compute correlation coefficient (r) for all response pairs
2. Examine frequency distribution for all $|r|$ to determine threshold for correlation pairs to retain; $|r| > 0.65$, here
3. Create clusters of mutually correlated pairs; each cluster represents one dimension
4. Select one response from each cluster to represent the dimension; we selected response with largest mean correlation that was not in another cluster*

Response Dimension	SA1-small (9 dimensions)	SA1-large (8 dimensions)	SA2-small (10 dimensions)	SA2-large (9 dimensions)
Cloud-wide Demand/Supply Ratio	*y1, y2,* ***y3****, y5, y6, y8, y9, y10, y13, y23, y24, y25, y29, y30, y32, y34, y36, y38*	*y1, y2,* ***y3****, y5, y6, y7, y8, y9, y10, y13, y23, y34, y25, y29, y30, y32, y33, y34, y36, y38*	*y1,* ***y2****, y3, y5, y6, y8, y9, y10, y11, y13, y14, y15, y23, y24, y25, y38*	*y1, y2, y3, y5, y6, y8, y9,* ***y23****, y24, y25, y38*
Cloud-wide Resource Usage	*y10, y11, y12, y13, y14,* ***y15***	*y10, y11, y12, y13, y14,* ***y15***	***y10****, y11, y12, y13, y14, y15*	***y10****, y11, y12, y13, y14, y15*
Variance in Cluster Load	*y16, y17, y18, y19,y20, y21,* ***y26****, y27*	*y16, y17, y18, y19,y20, y21,* ***y26****, y27*	*y16, y18, y19, y20, y21, y26,* ***y27*** ***y17*** (Mem. Util)	*y16, y17, y18,* ***y19****,y20, y21, y26, y27*
Mix of VM Types	*y34,* ***y35*** (WS) ***y31*** (MS)	***y31*** (MS)	*y12, y14, y15, y30, y31, y33, y34, y35,* ***y36***	*y14, y15, y30,* ***y31****, y33, y34, y35* *y15,* ***y36*** (DS)
Number of VMs	*y29,* ***y37***	***y37***	*y29,* ***y37***	***y29***
User Arrival Rate	***y4***	***y4***	***y4***	***y4****, y37*
Reallocation Rate	***y7****, y22*	*y7,* ***y22***	***y7*** (cluster) ***y22*** (node)	*y7,* ***y22***
Variance in Choice of Cluster	***y28***	***y28***	***y28***	***y28***

*Not possible for cloud-wide resource usage in SA2-small, so we selected response with highest mean correlation.

Main Effects Analysis (MEA) Identifies Significant Influence of Input Parameters on Response Variables

We applied MEA to response variables selected using CAC – this example is ***y15*** (NIC Count Load) for **experiment SA1-small**



Most significant parameters determined through MEA of the responses selected using CAC

We computed percent of responses influenced (Ψ) for each parameter, weighting $p < 0.05$ at ½ and $p < 0.01$ at 1:

$$\Psi = (|\{y \mid p < 0.01\}| + \tfrac{1}{2}\,|\{y \mid p < 0.05\}|) / |\{y\}| \times 100$$

Computed average Ψ for each parameter, weighting experiment Ψ by number of repetitions

		Input Parameter										
Experiment	Weight	*x1*	*x2*	*x3*	*x4*	*x5*	*x6*	*x7*	*x8*	*x9*	*x10*	*x11*
SA1 small	6/14	1	57	22	11	44	29	30	12	0	1	0
SA1 large	1/14	0	69	13	25	44	56	31	25	0	13	0
SA2 small	6/14	2	73	38	10	45	62	10	17	1	0	0
SA2 large	1/14	0	56	50	11	39	56	6	11	0	0	0
Avg. Ψ	Est.	1	65	30	12	44	47	20	15	0	1	0

green = major influence; yellow = modest influence; orange = minor influence; gray = no influence

Most significant parameters: *x2* (# users), *x5* (# clusters), and *x6* (# nodes/cluster)
Moderately influential parameters: *x3* (user types) and *x7* (platform types)
Somewhat influential parameters: *x4* (user hold time) and *x8* (cluster-selection algorithm)
No influence : *x1* (measurement interval), *x9* (node-selection algorithm), *x10* (geo-distribution of cloud components), and *x11* (packet loss prob.)

Ongoing Work

Currently conducting an experiment to compare 18 resource allocation heuristics for on-demand IaaS Clouds

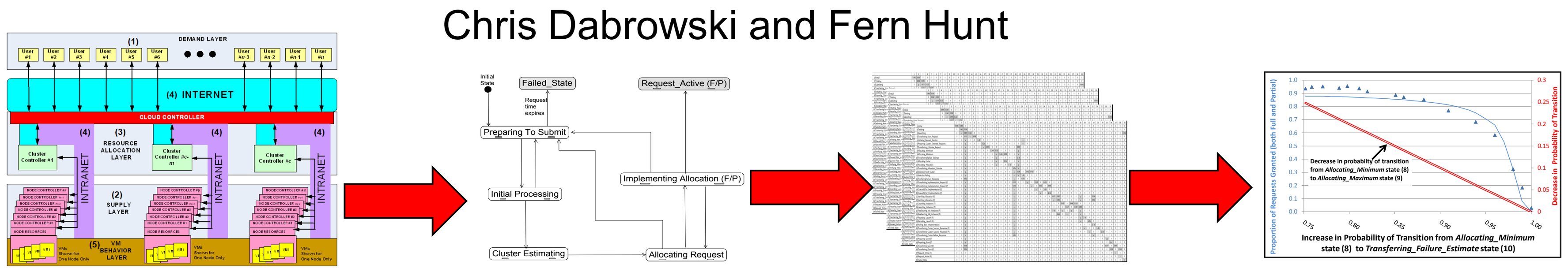
Cluster Selection	Node Selection
Least Full First	First Fit
	Next Fit
Percent Allocated	Tag & Pack
	Random
Random	Least Full First
	Most Full First
3	x 6 = 18

Experiment design is "Resolution VI" 2^{5-1} OFF, requiring simulating each of the 18 heuristics under 32 conditions (i.e., 576 total simulations)

Simulations are completed, data collected and summarized. Data analysis ongoing.

IDENTIFY FAILURE SCENARIOS IN CLOUD SYSTEMS USING MARKOV CHAIN ANALYSIS

Chris Dabrowski and Fern Hunt



Problem: Identifying failure scenarios in distributed systems such as clouds is critical to understanding areas where performance may degrade. However, potential failure scenarios may be numerous and difficult to find.

Objective: To perturb Discrete Time Markov Chains (DTMCs) of cloud system behavior to identify potential failure scenarios more quickly than through detailed large-scale simulation or use of test beds.

Steps (describing today):

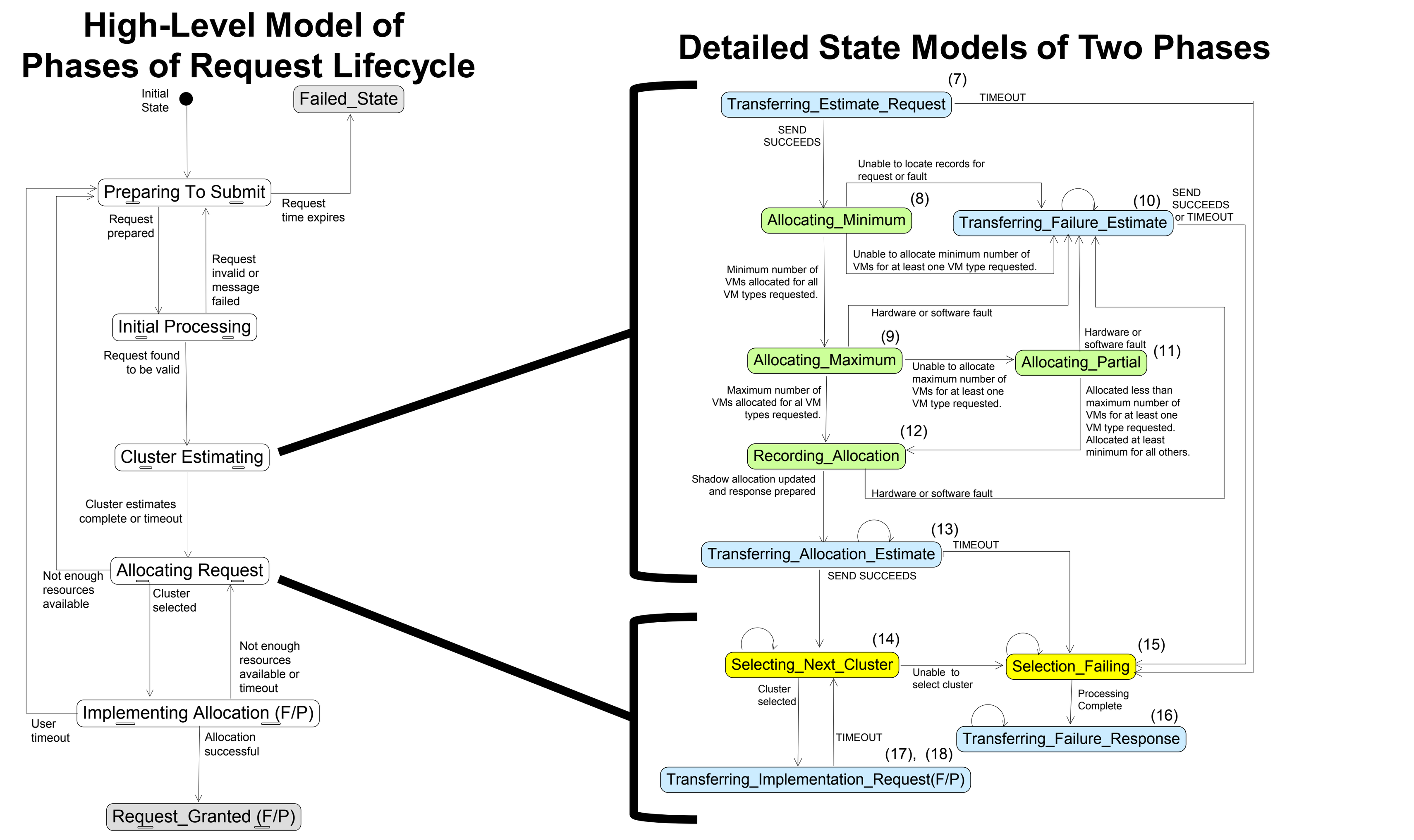
(1) Using *Koala* as proxy for real-world cloud, develop detailed state model of cloud behavior and convert to time-inhomogeneous DTMC.

(2) Find minimal s-t cut sets in a directed graph of cloud DTMC to identify critical state transitions that break paths to desirable system goal states.

(3) Perturb critical state transitions to describe potential failure scenarios, create predictive performance curves, and find performance thresholds.

State Model of Resource Request in Cloud

A detailed representation of states that a cloud system (*Koala*) may enter under normal and failure conditions, shown for two five major phases.

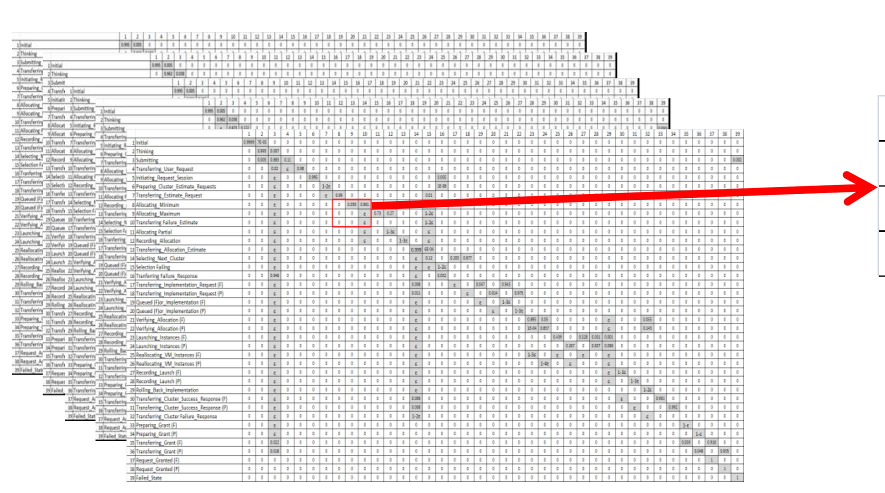


Creating a Discrete Time Markov Chain

- Observe *Koala* (as proxy for real-world system) to derive set of transition probability matrices (TPMs) that describe probabilities of transition between states over different time periods →forms a time-inhomogeneous DTMC.
- Generated1000 time period TPMs of 3600 s each.

$$p_{ij} = \frac{f_{ij}}{\sum_{k=1}^{n} f_{ik}}$$

Given states s_i, s_j, $i,j = 1…n$ where n=39, p_{ij}, is the probability of transitioning from state i to state j, written as $s_i \rightarrow s_j$. This probability is estimated by calculating the frequency of $s_i \rightarrow s_j$, or f_{ij}, divided by the sum of the frequencies of s_i to all other states.



		8	9	10
8	Allocating_Minimum	0	0.264	0.736
9	Allocating_Maximum	0	0	ε
10	Transferring Failure_Estimate	0	0	ε

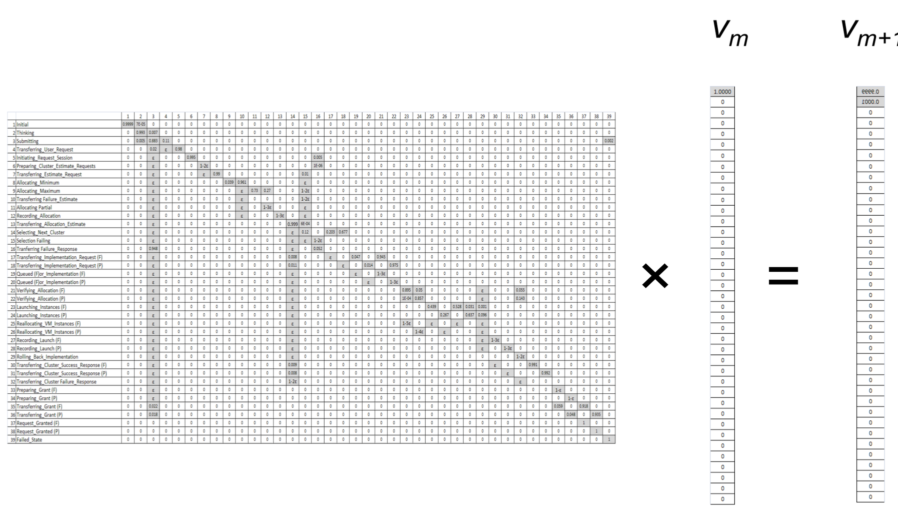
Using the DTMC to simulate large-scale system (*Koala*) behavior

- Markov chains can emulate *Koala* to capture high-level system behavior, but in two orders of magnitude less computational time.

To evolve system state in discrete time steps, multiply state vector v_m (at time step m) by the TPM, Q^{tp}, for the applicable time period tp to produce a new system state vector v_{m+1},

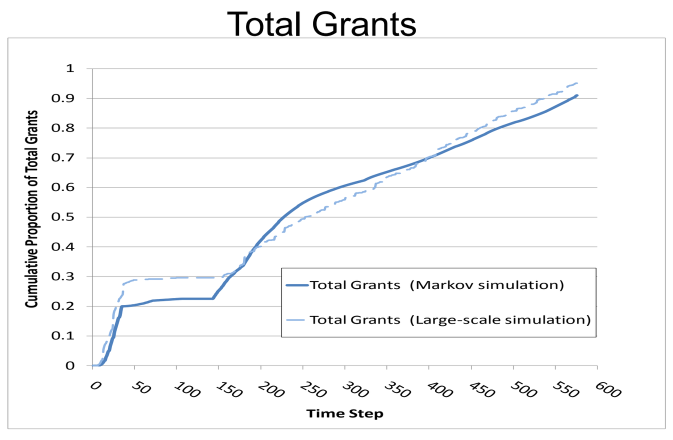
$$(Q^{tp})^T * v_m = v_{m+1}, \text{ where } tp = \text{integral value } (m/S) + 1$$

where T indicates a matrix transpose.

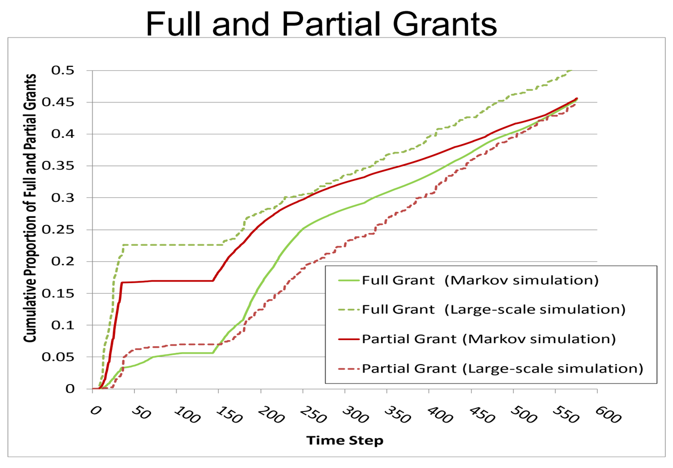


Repeated for 576 time steps in 16 hour simulated period, one time period per hour.

Total Grants

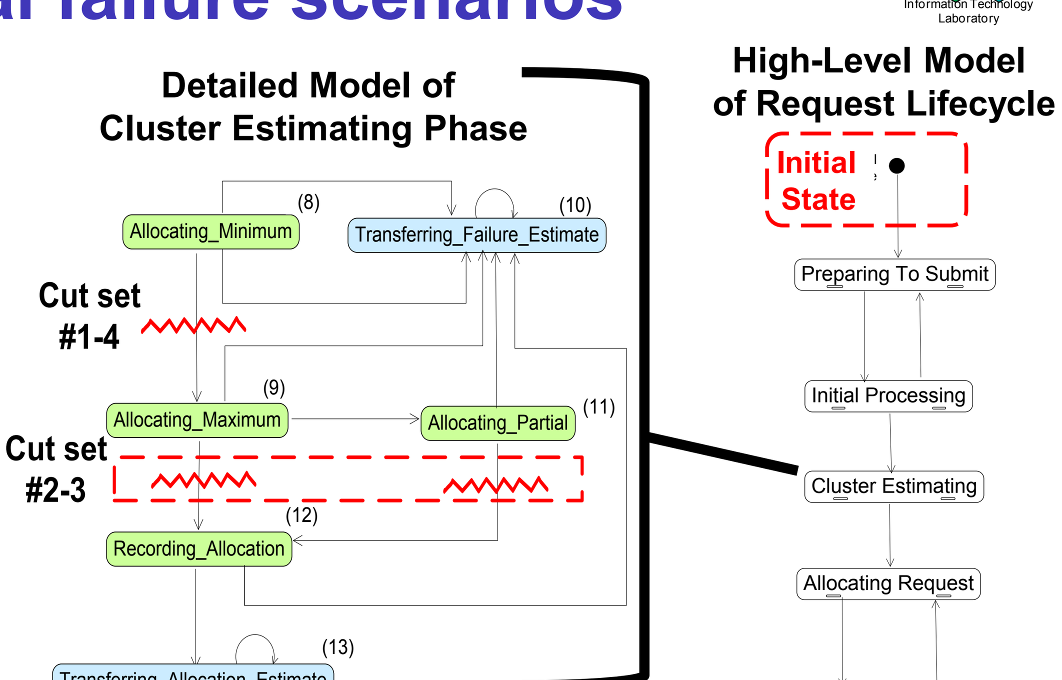


Full and Partial Grants



Using minimal s-t cut set analysis to find potential failure scenarios

- In a directed graph of the *Koala* DTMC, minimal s-t cut sets consist of *critical state transitions*, which if removed, disconnect all paths to absorbing *Requests_Granted (F/P)* state.
- Applying algorithm to find minimal s-t cut sets* to the *Koala* DTMC resulted in 159 cut sets. Examples of one and two-transition cut sets are shown.



Both cut sets disconnect all paths from *Initial State* to *Request _Granted (F/P)* absorbing states.

One-transition cut sets

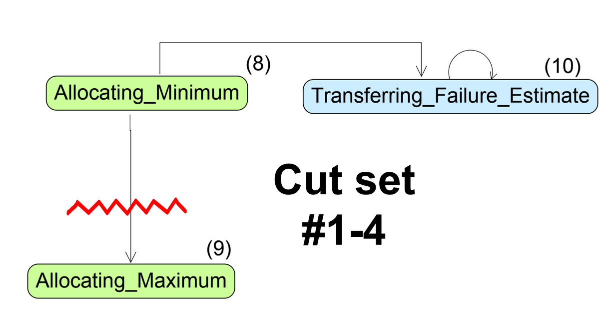
	Set of member transitions from Fig. 3	Total Probabilty
1-1	{1, 2}	0.001
1-2	{2, 3}	0.025
1-3	{3, 4}	0.124
1-4	{8, 9}	0.264
.....	....	....
1-10	{12, 13}	1.000

Two-transition cut sets

	Set of member transitions from Fig. 3	Number of From States	Total Probabilty
2-1	{14, 17} {14, 18}	1	0.895
2-2	{9, 11} {9, 12}	1	1.000
2-3	{9, 12} {11, 12}	2	1.395
.....	....	..	....
2-23	{33, 35} {34, 36}	2	2.000

*Provan S., and Ball M., 1984, "Computing Network Reliability in Time Polynomial in the Number of Cuts," *Operations Research*, 32(3), pp. 516–526.

Perturbing state transitions in a cut set to predict system behavior in failure scenario (1)

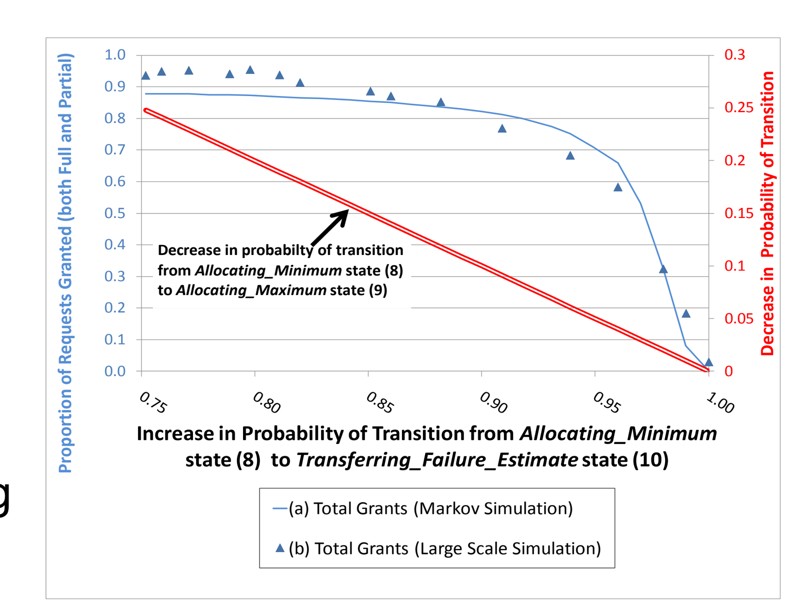


- Cut set #1-4 could relate to a scenario in which software or hardware failures make resource databases inaccessible, preventing clusters from computing minimum allocation estimates. Instead, clusters return failure estimates to the cloud controller.

Portions of TPM perturbed

		8	9	10
8	Allocating_Minimum	0	0.248	0.752
9	Allocating_Maximum	0	0	ε
10	Transferring Failure_Estimate	0	0	ε

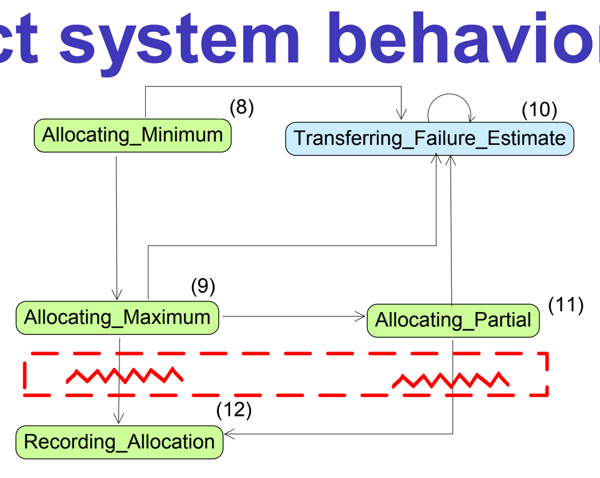
- Raise probabiltiy of *Allocating_Minimum* → *Transferring_Failure_Estimate*:TPM element {8, 10}
- Lower probablity of *Allocating_Minimum* → *Allocating_Maximum:* TPM elements {8, 9}.



Decline in total requests granted (Full and Partial) due to cluster estimation failure:
(a) As estimated by perturbing the DTMC; and
(b) As computed in *Koala* large-scale simulation.

Blue curves show the resulting decrease in requests granted as estimated using the DTMC and as actually occurred in the *Koala* large-scale simulation. These curves are plotted against the left vertical axis. The right vertical axis provides units for the decrease in probability of the state transition.

Perturbing state transitions in a cut set to predict system behavior in failure scenario (2)



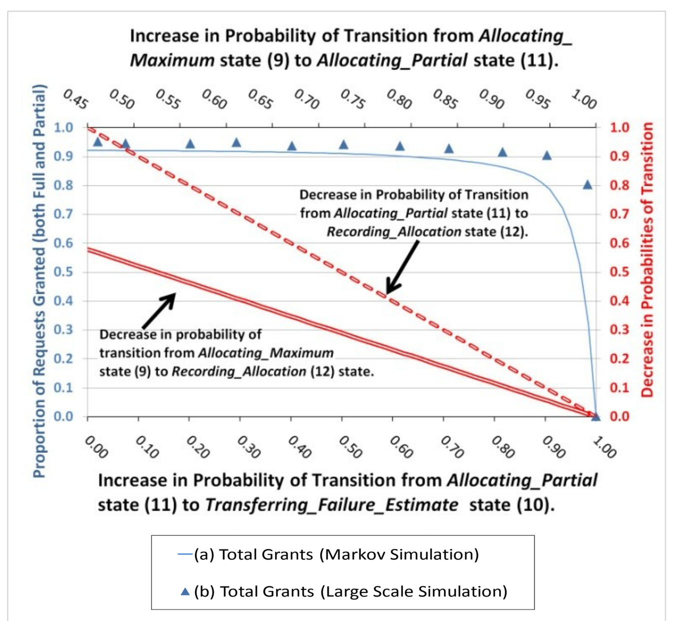
- Cut set #2-3 could relate to a failure scenario in which viruses or other faults cause widespread software process failures in clusters, which prevent completion of cluster allocation estimation computations. Instead, clusters return failure estimates to the controller.

Portions of TPM perturbed

		9	10	11	12
9	Allocating_Maximum	0	ε	0.464	0.536
10	Transferring Failure_Estimate	0	ε	0.000	0.000
11	Allocating Partial	0	ε	0.000	1-3ε
12	Recording_Allocation	0	ε	0.000	0.000

•Raise *Allocating_ Maximum* → *Allocating_Partial*: TPM element {9, 11}
•Lower*Allocating_Maximum* → *Recording_ Allocation* (TPM element {9, 12})

•Raise *Allocating_Partial*→*Transferring_Failure_Estimate:* TPM element {11, 10}
•Lower *Allocating_Partial* → *Recording_Allocation*: TPM element {11, 12}



Decline in total requests granted (Full and Partial) due to cluster estimation failure:
(a) As estimated by perturbing the DTMC; and
(b) As computed by *Koala* large-scale simulation.

Blue curves show the resulting decrease in requests granted as estimated using the DTMC and as actually occurred in the *Koala* large-scale simulation. These curves are plotted against the left vertical axis. The right vertical axis provides units for the decrease in probability of the state transition.

Ongoing Work

Apply methodology to larger problems and determine scalability

- Current model consists of 39 states and 139 transitions
- Includes user, cloud controller, and cluster behavior, but not node behavior or actual use of VMs

Apply methodology to different types of failure scenarios

For more information, see: Identifying Failure Scenarios in Complex Systems by Perturbing Markov Chain Models ,by Christopher Dabrowski and Fern Hunt , submitted to *ASME 2011 PVPD Conference*